EXAMINER'S AMENDMENT AND REASONS FOR ALLOWANCE

An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given by applicant's agent or attorney of record, Beatrice L. Koempel-Thomas (Reg. No. 58213), via a telephonic examiner-initiated interview on 6/4/2009, and in an email communication received on 6/10/2009 (see attached email) wherein amendments to all independent claims were discussed. The purpose of the amendment is to clarify the invention and to distinguish the invention from the prior art of record.

The application has been amended as follows:

Please amend the claims as indicated in the complete listing of the claims attached to the end of this document. - See pages 9-23.

The following is the Examiner's statement of reasons for allowance:

The prior art of record does not disclose nor render obvious the claimed subject matter as recited in amended Claims 1-39, 41-48, and 50-64 (to be renumbered claims 1-62 respectively), especially regarding the amended independent Claims 1, 38, 45, 54,

58 and 63 (to be renumbered 1, 38, 44, 52, 56 and 61, respectively) and their limitations comprising:

Claim 1 (to be renumbered Claim 1):

a contract component that facilitates stating a behavioral contract defined according to an object oriented programming language on an asynchronous service interface between a client interface and a service interface; and

a model extraction component that automatically extracts, on an individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service.

Claim 38 (to be renumbered Claim 38):

an extraction component that automatically extracts a client model from a client interface and a service model from a service interface on an individual function level, such that a model checking component may systematically explore all of the possible interactions between the client and the service;

a relator component that creates a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client model and the service model;

Claim 45 (to be renumbered Claim 44):

creating a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client interface and the service interface;

automatically extracting via a model extraction component, on an individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service.

Claim 54 (to be renumbered Claim 52):

automatically extracting a client model from the client interface and a service model from the service interface on an individual function level, such that a model checking component may systematically explore all of the possible interactions between the client and the service;

creating a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client model and the service model;

Claim 58 (to be renumbered Claim 56):

a contract component that creates a contract state defined according to an object oriented programming language between a client state and a service state;

an extraction component that automatically extracts a client model from the client state, a service model from the service state, and an interface model from the contract state on an individual function level, such that a conformance checking component may systematically explore all of the possible interactions between the client and the service;

Claim 63 (to be renumbered Claim 61):

means for automatically extracting a client model from the client interface and a service model from the service interface on an individual function level, such that a means for checking may systematically explore all of the possible interactions between the client and the service;

means for creating a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client model and the service model;

in the context of their remaining claim limitations and as best illustrated by figures 1 and 2, and pages 3, 16 and 17 of the as-filed specification.

The closest prior art of record, Rehof et al., (USPGPUB 2003/0204570, hereinafter 'Rehof'), and Beugnard et al., "Making Components Contract Aware", IEEE, 1999, hereinafter 'Beugnard', do not separately anticipate nor jointly or in combination

with other prior art disclosures render obvious the aforementioned limitations of the pending independent claims.

Regarding Rehof, Examiner agrees with Applicants' remarks comprising:

> Regarding independent claim 38, Rehof et al. fails to disclose "a conformance checking component that checks that the client interface obeys the behavioral relationship, and the service interface properly implements the behavioral relationship," as recited in the subject claim. The Office Action again relies on Rehof et al. at par. [0009] for the noted features. However, as discussed supra, Rehof et al. describes one-sided operations relating only to a particular implementation of a message passing program module, and is silent ***regarding checking conformance to a relationship between a client interface and a service interface***.
>
> As to independent claim 45, along similar lines, Rehof et al. does not disclose "checking to ensure that the client interface obeys the behavioral relationship and the service interface properly implements the behavioral relationship," as recited.

(Remarks, pg. 12, emphasis original), and notes that similar recitations are present in the amended independent claims. Accordingly, Rehof does not anticipate the invention(s) set forth in the currently amended claims.

Beugnard discloses a general model of software contracts and describes mechanisms for enabling legacy components to become contract-aware components. (Beugnard, pg. 38). Beugnard teaches the use of "Behavioral Contracts" such that the 'design-by-contract' software engineering method allows "developers to specify precisely every consistency condition that could go wrong, and to assign explicitly the responsibility of its enforcement to either the routine caller (the client) or the routine implementation (the contractor)." (Beugnard, pg. 39). However, Beugnard does not disclose nor in combination with any other reference render obvious the definition of

contracts in terms of an object oriented language, nor a model extraction component that automatically extracts, on an individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service, as recited analogously in all independent claims and as set forth in the as-filed specification. Therefore, the cited references, taken alone or in view of any other recorded reference, do not anticipate nor render obvious the invention of the amended independent claims, specifically regarding the aforementioned claim limitations. Accordingly, the pending claims are patentable over the prior art of record.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to RYAN D. COYER whose telephone number is (571) 270-5306, and whose fax number is (571) 270-6306.. The examiner can normally be reached via phone on Mon-Thurs, 9a-7p. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen, can be reached on (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/RYAN D COYER/
Examiner, Art Unit 2191

/Wei Y Zhen/
Supervisory Patent Examiner, Art Unit 2191

PLEASE AMEND THE CLAIMS AS FOLLOWS:

1. (Currently Amended) A system that facilitates asynchronous programming, comprising:

at least one processor that executes the following computer executable components stored on at least one computer readable medium:

a contract component that facilitates stating a behavioral contract defined according to an object oriented programming language on an asynchronous service interface between a client interface and a service interface; ~~and~~

a conformance checking component that checks that ~~at least one of~~ the client interface and the service interface conform to the contract defined therebetween; and

a model extraction component that automatically extracts, on an individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service.

2. (Original) The system of claim 1, the contract defines an order in which clients of the service interface are invoked.

3. (Original) The system of claim 2, the checking component checks that the

clients handle both normal and exceptional results.

4. (Currently Amended) The system of claim 1, at ~~least one of~~ wherein the asynchronous service interface and the contract are defined according to an objected oriented program language.

5. (Original) The system of claim 1, the checking by the checking component is modular wherein the contract is specified on an interface boundary of the client interface and the service interface and, each of the client interface and the service interface is checked independently, and each method is checked separately.

6. (Original) The system of claim 1, the checking component further comprises a model extraction component that automatically extracts a behavior model that is fed to a model checker component.

7. (Original) The system of claim 6, the extracted model is sound in the presence of aliasing.

8. (Original) The system of claim 1, further comprising a construct in the form of an asynchronous call that returns a future-like handle to an eventual result of the call.

9. **(Original)** The system of claim 1, further comprising a construct in the form of a synch statement.

10. **(Original)** The system of claim 1, further comprising a construct that states the behavioral contract on the service interface.

11. **(Original)** The system of claim 1, further comprising a construct that is a transaction directive that delineates the scope of conversations by clients.

12. **(Original)** The system of claim 1, further comprising a construct that is a tracked type qualifier used by the checking component.

13. **(Original)** The system of claim 1, further comprising a procedure, which procedure is a method that is called at least one of synchronously and asynchronously.

14. **(Original)** The system of claim 13, the asynchronous call executes concurrently with its caller by completing immediately, while the method that is invoked executes.

15. (Original) The system of claim 14, the asynchronous call completes by returning a value.

16. (Original) The system of claim 15, the value is used explicitly to synchronize the asynchronous call with the value.

17. (Original) The system of claim 15, the value is a first-class type that can be at least one of stored, passed as an argument, and returned as a result.

18. (Original) The system of claim 17, the result returned is explicitly retrieved through a join statement.

19. (Original) The system of claim 15, the value is an object in one of three states that include an undefined state, normal state, and exceptional state.

20. (Original) The system of claim 1,the contract is a source-level contract that facilitates statically detecting asynchronous message passing errors.

21. (Original) The system of claim 1, the contract expresses stateful protocols that govern interactions between a client associated with the client interface and a service associated with the service interface.

22. **(Original)** The system of claim 1, the service interface is associated with a service contract whose language models the effects of multiple clients that concurrently access the service interface.

23. **(Original)** The system of claim 1, the checking component checks that services and clients obey the contract associated with the service interface.

24. **(Original)** The system of claim 1, the checking component uses a transaction directive to delimit a scope of an instance of a concurrent interaction between a client and a set of service instances.

25. **(Original)** The system of claim 24, the transaction directive specifies a correlation variable that uniquely identifies the instance of the concurrent interaction between the client and the set of service instances.

26. **(Original)** The system of claim 1, the checking component is modular in that to check for the client conformance and the service conformance, only the associated contract needs to be referenced.

27. **(Original)** The system of claim 1, the checking component considers a

method as a function of an object to which it belongs.

28. (Original) The system of claim 1, the checking component relates a contract state, a client state, and a service state.

29. (Original) The system of claim 28, the checking component hrther comprises an extraction component that extracts a client model, service model, and interface model in order to relate the contract state, client state, and service state.

30. (Original) The system of claim 28, the checking component further comprises an extraction component that uses a region to model relevant data, which relevant data is that data which is directly relevant to the contract state.

31. (Original) The system of claim 30, the region is polymorphic.

32. (Original) The system of claim 30, the region is partial such that only relevant data is directly assigned to a region type.

33. (Original) The system of claim 1, the checking component hrther comprises an extraction component that facilitates model reduction by utilizing only

necessary statements.

34. (Original) The system of claim 1, the checking component further comprises an extraction component that utilizes region-and-effect analysis to extract a model.

35. (Original) The system of claim 1, the checking component hrther comprises an extraction component that utilizes type-and-effect inference and source-level tracked types.

36. (Original) The system of claim 1, the contract is programmer-specified to separately check client and server code.

37. (Original) A computer according to the system of claim 1.

38. (Currently Amended) A system that facilitates asynchronous programming, comprising:

at least one processor that executes the following computer executable components stored on at least one computer readable medium:

an extraction component that automatically extracts a client model from a client interface and a service model from a service interface on an individual function level,

such that a model checking component may systematically explore all of the possible interactions between the client and the service;

a relator component that creates a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client model and the service model; and

a conformance checking component that checks that the client interface obeys the behavioral ~~relationship~~ contract, and the service interface properly implements the behavioral ~~relationship~~ contract.

39. (Currently Amended) The system of claim 38, the behavioral ~~relationship~~ contract expressed in terms of at least one of a future, a join on the future, and asynchronous function calls.

40. (Canceled)

41. (Currently Amended) The system of claim 38, the conformance checking component checks that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results.

42. (Original) The system of claim 38, the extraction component creates the client model that defines relevant data for interaction with the service interface.

43. (Original) The system of claim 38, the extraction component creates the service model that defines relevant data for interaction with the client interface.

44. (Currently Amended) The system of claim 38, the relator component checks the client model and the service model such that contract descriptions of the behavioral ~~relationship~~ contract are defined between the client model and the service model.

45. (Currently Amended) A method of asynchronous programming, comprising:

employing a processor to execute computer readable instructions stored on a computer readable medium to perform the following acts:

receiving a client interface and a service interface;

creating a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client interface and the service interface; ~~and~~

checking to ensure that the client interface obeys the behavioral ~~relationship~~ contract and the service interface properly implements the behavioral ~~relationship~~ contract; and

automatically extracting via a model extraction component, on an

individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service.

46. (Currently Amended) The method of claim 45, the behavioral ~~relationship~~ contract is expressed in terms of at least one of a future, a join on the future, and asynchronous function calls.

47. (Original) The method of claim 45, the behavioral relationship is a contract the terms of which are enforced by a modular checking component.

48. (Original) The method of claim 47, the modular checking component checks that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results.

49. (Canceled)

50. (Currently Amended) The method of claim [[49]] 45, the model extraction component creating the client model that defines relevant data for interaction with the service interface.

51. (Currently Amended) The method of claim [[49]] 45, the model extraction component creates the service model that defines relevant data for interaction with the client interface.

52. (Currently Amended) The method of claim [[49]] 45, further comprising performing model checking of the client model and the service model with a model checker such that contract descriptions are defined between the client model and the service model.

53. (Currently Amended) The method of claim 45, further comprising automatically extracting the behavioral ~~relationship~~ contract.

54. (Currently Amended) A method of asynchronous programming, comprising:

employing a processor to execute computer readable instructions stored on a computer readable medium to perform the following acts:

receiving a client interface and a service interface;

automatically extracting a client model from the client interface and a service model from the service interface on an individual function level, such that a model checking component may systematically explore all of the possible interactions between the client and the service;

creating a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client model and the service model; and

enforcing the behavioral ~~relationship~~ contract between the client interface and the service interface.

55. (Currently Amended) The method of claim 54, the behavioral ~~relationship is~~ contract comprises a contract expressed by constructs in terms of at least one of a future, a join on the future, and asynchronous function calls, and the terms of which are enforced by a modular checking component.

56. (Original) The method of claim 54, further comprising checking that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results.

57. (Currently Amended) The method of claim 54, wherein ~~further comprising: extracting with a model extraction component a client model from the client interface,~~ the client model defines relevant data for interaction with the service interface for the client interface,[[;]] and ~~extracting with the model extraction component a service model from the service interface,~~ the service model defines relevant data for interaction with the client interface.

58. (Currently Amended) An article of manufacture tangibly embodying program instructions for execution by a processing unit, the processing unit retrieving the instructions from a computer-readable memory to facilitate concurrent execution of the instructions, the product comprising:

a contract component that creates a contract state defined according to an object oriented programming language between a client state and a service state;

an extraction component that automatically extracts a client model from the client state, a service model from the service state, and an interface model from the contract state on an individual function level, such that a conformance checking component may systematically explore all of the possible interactions between the client and the service;

and [[a]] the conformance checking component ~~that~~ checks that the client model obeys the interface model, and the service model properly implements the contract model.

59. (Previously Presented) The article of claim 58, the client model, service model, and interface model contain only respective communication actions and operations that are relevant for maintaining the state of a communication protocol.

60. (Previously Presented) The article of claim 59, the relevant operations are represented by the use of regions.

61. (Previously Presented) The article of claim 58, the interface model includes one or more effect processes for each method of the interface state.

62. (Previously Presented) The article of claim 61, the effect process models a method of the interface state according to an effect a call has on the contract state, a future created by an asynchronous call to the method, and futures passed into the method.

63. (Currently Amended) A system of asynchronous programming, comprising one or more computing devices configured with:

means for receiving a client interface and a service interface;

means for automatically extracting a client model from the client interface and a service model from the service interface on an individual function level, such that a means for checking may systematically explore all of the possible interactions between the client and the service;

means for creating a behavioral ~~relationship~~ contract defined according to an object oriented programming language between the client model and the service model; and

the means for checking to ensure that the client interface obeys the behavioral ~~relationship~~ contract and the service interface properly implements the behavioral ~~relationship~~ contract.

64. (Previously Presented) The system of claim 18, the first class type is introduced by one of a local declaration or a field member declaration.

/Wei Y Zhen/

Supervisory Patent Examiner, Art Unit 2191